실습으로 배우는 AWS

04 AWS 네트워킹 활용하기

/* elice */

목차

01

AWS 네트워킹 1 : 네트워킹 기본

/* elice */

네트워킹 기초

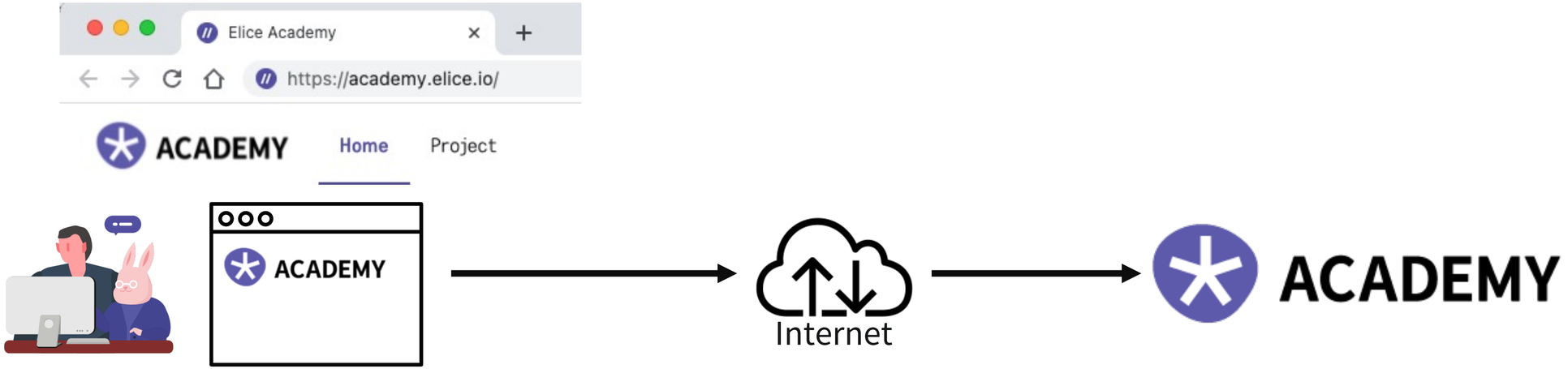
Client

요청 하는 주체

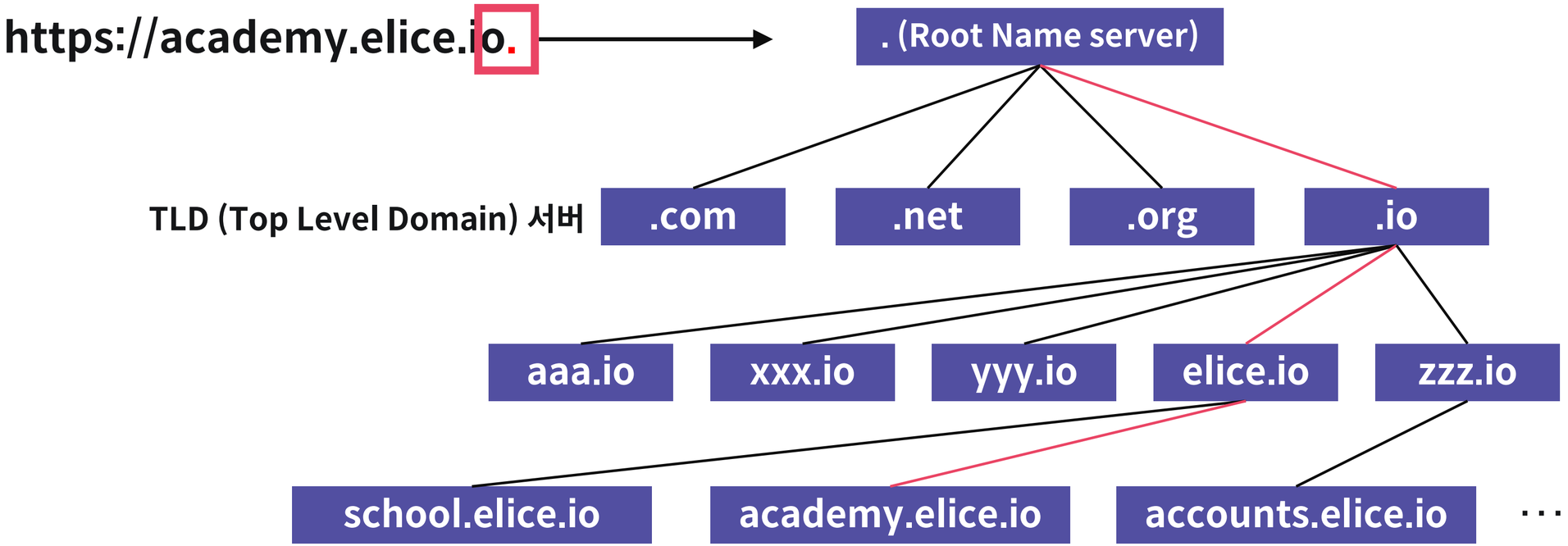
Server

요청을 수행하는 주체

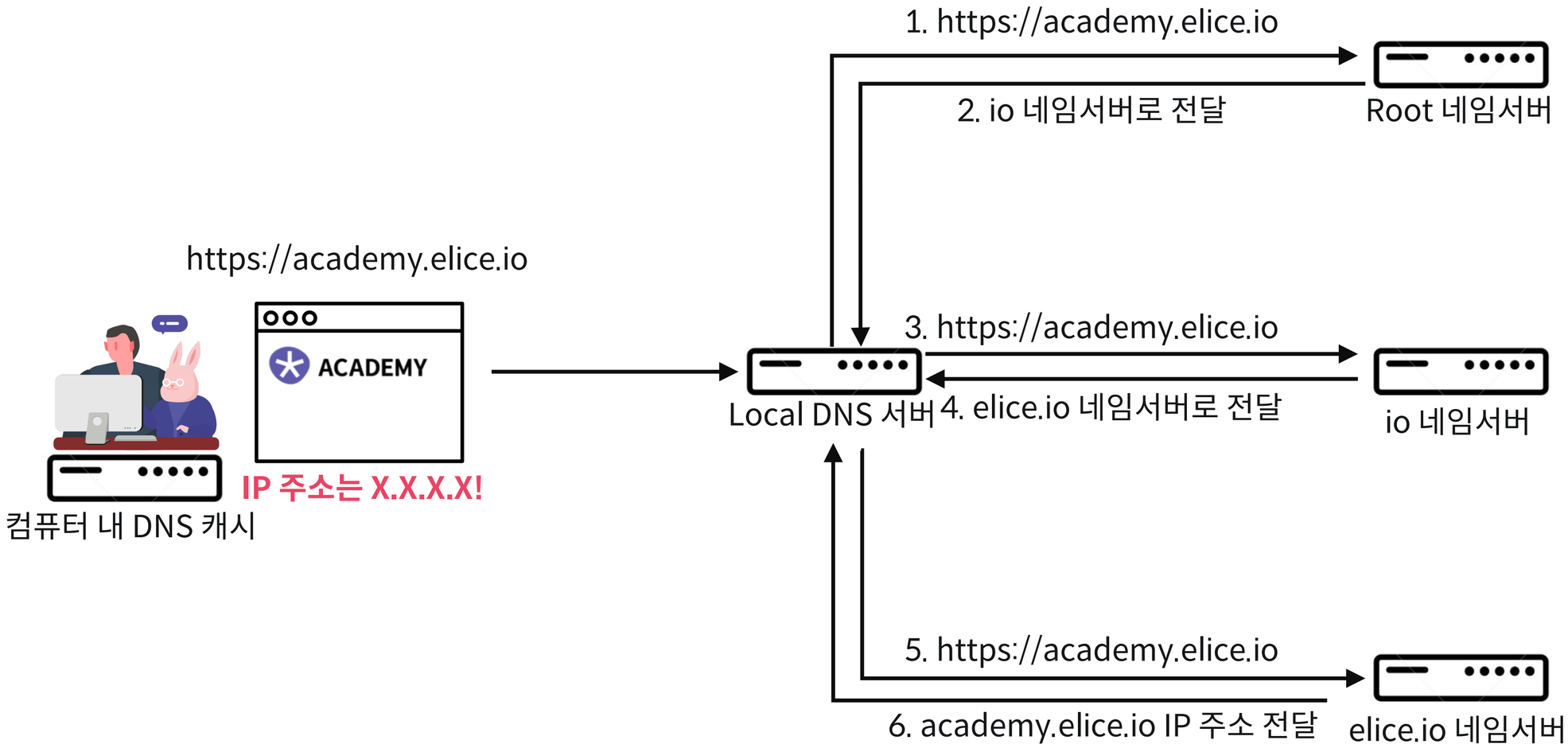
엘리스 아카데미에 접속하면 내부적으로는 무슨 일이 생길까요?



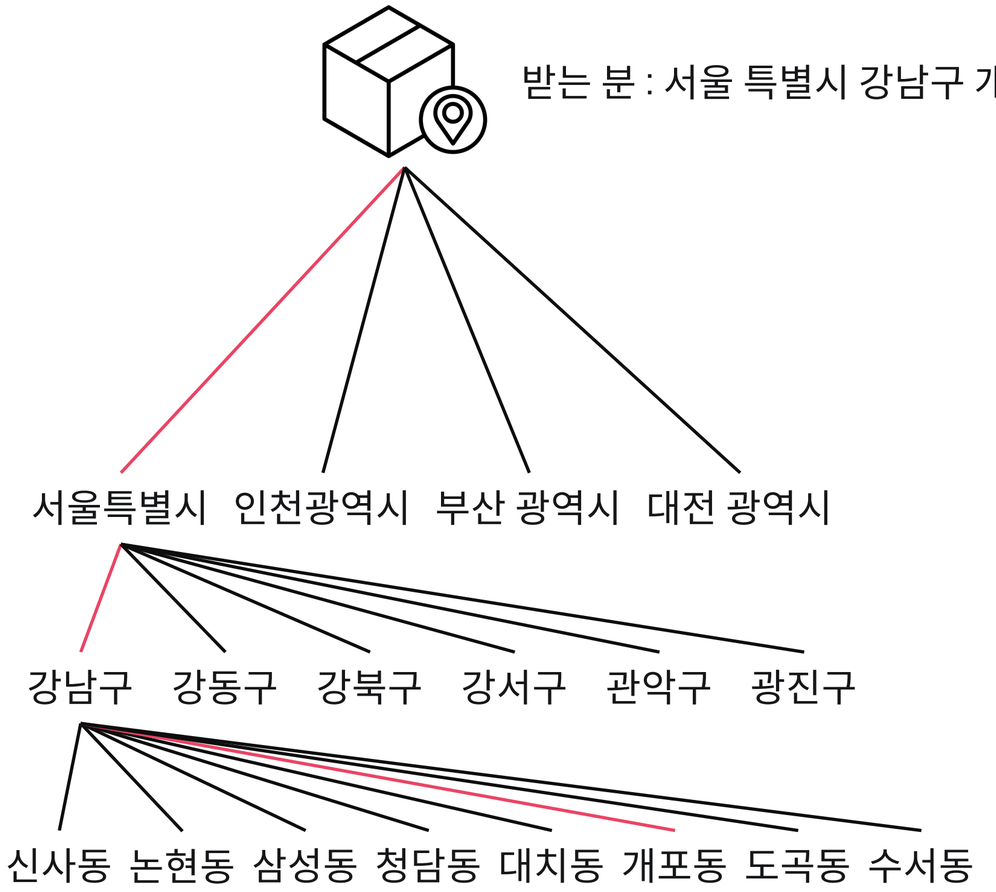
엘리스 아카데미에 접속하면 내부적으로는 무슨 일이 생길까요?

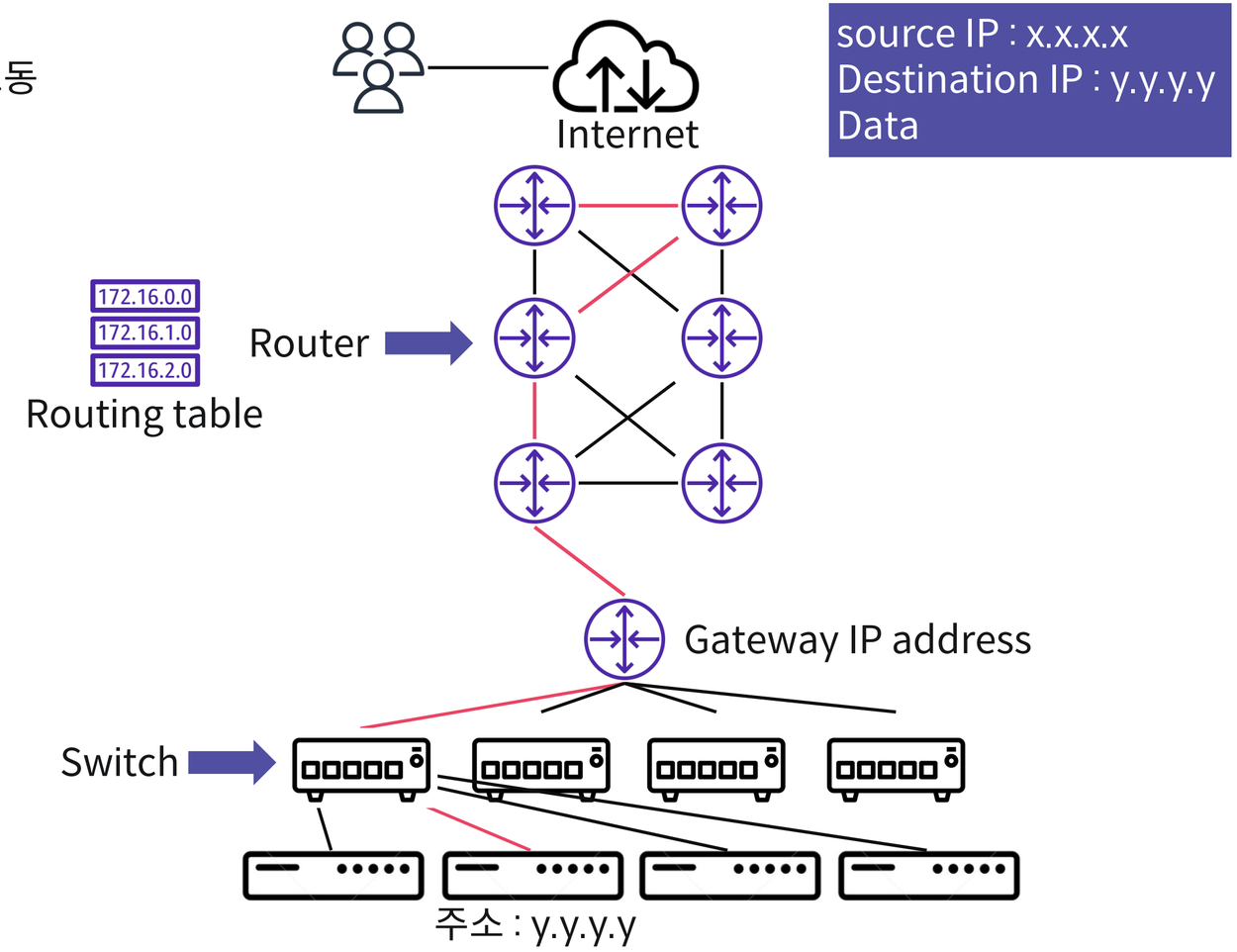


DNS 동작 방식

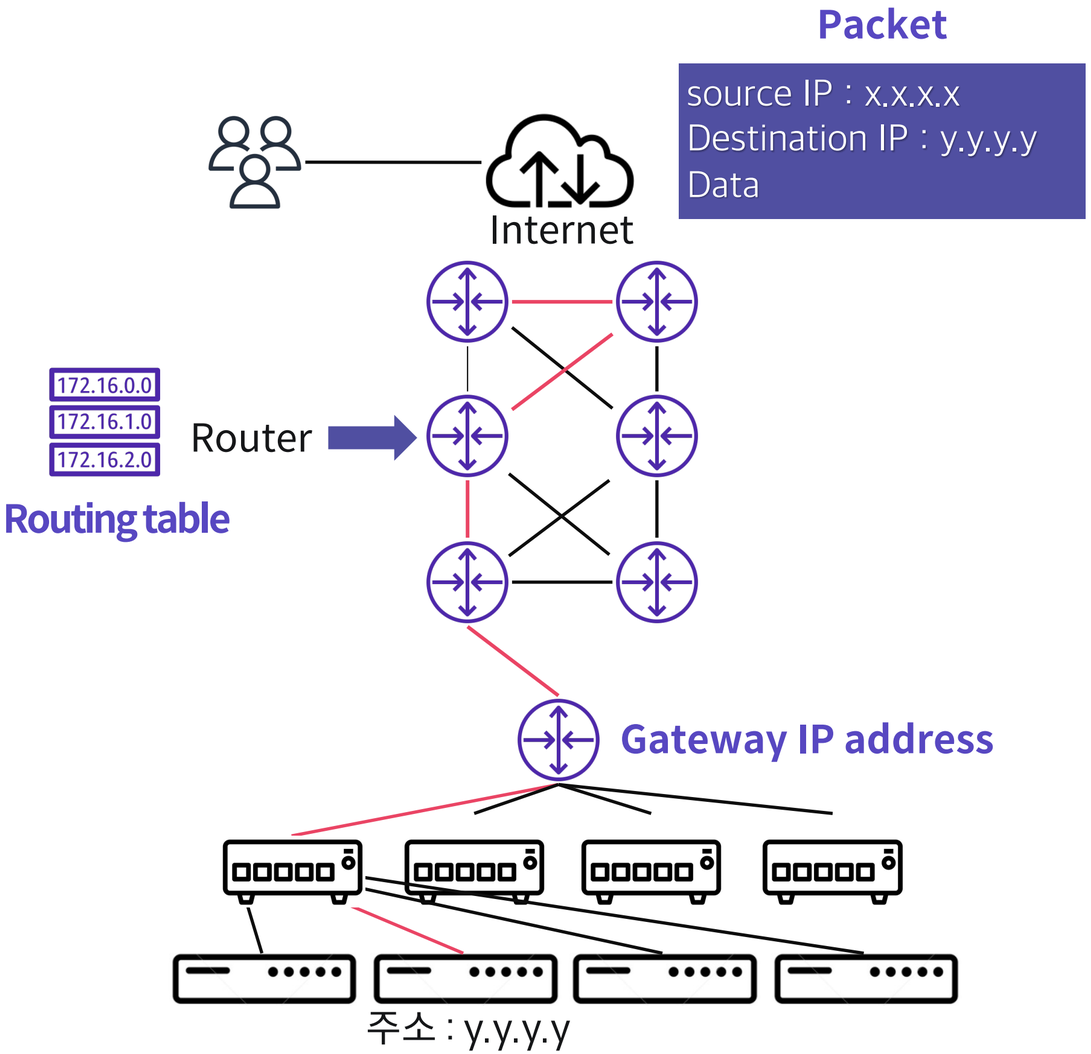

1. https://academy.elice.io
2. io 네임서버로 전달
Root 네임서버
https://academy.elice.io
ACADEMY
3. https://academy.elice.io
Local DNS 서버
4. elice.io 네임서버로 전달
io 네임서버
IP 주소는 X.X.X.X!
컴퓨터 내 DNS 캐시
5. https://academy.elice.io
6. academy.elice.io IP 주소 전달
elice.io 네임서버

IP 동작 방식 – vs 주소 시스템





IP 동작 방식

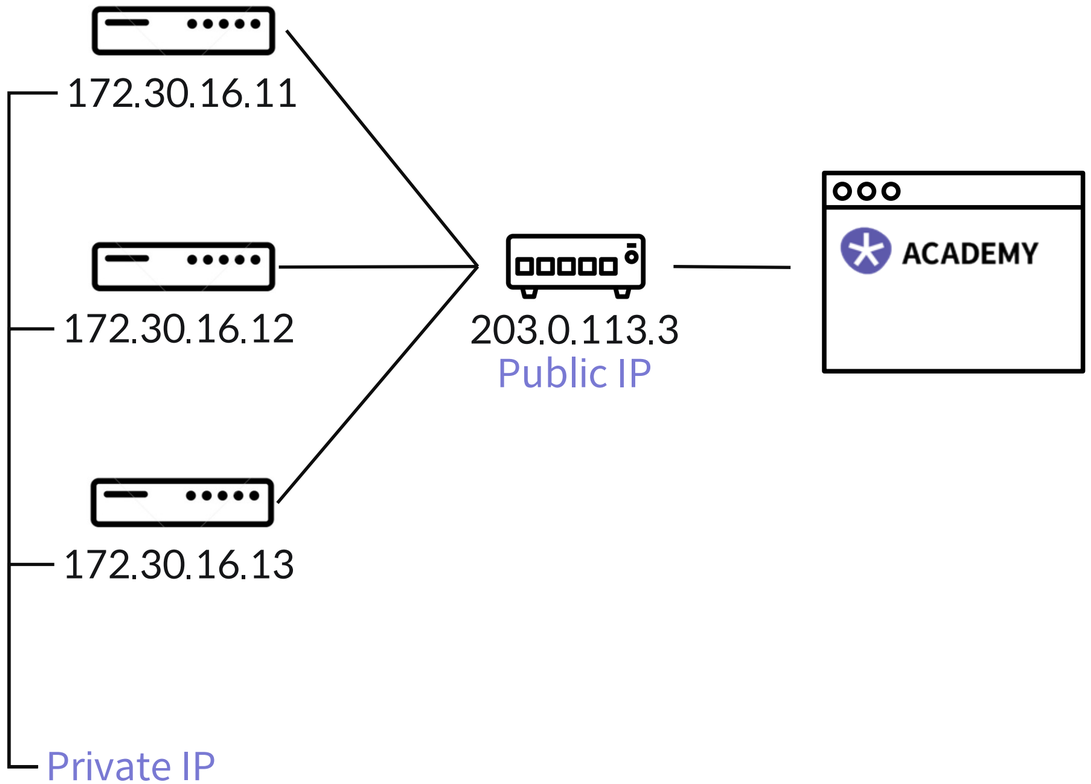


IP 와 Routing 동작 방식의 기본 규칙

1. 전달하려는 **패킷**에는 Source IP와 Destination IP를 포함 한다.
2. 전송하고자 하는 주소가 동일 IP 대역 내에 없으면 **Gateway IP**를 가진 라우터로 패킷을 보낸다.
3. 각각의 라우터는 패킷을 보낼 다음 홉 (라우터)의 주소를 알고 있다 = **Routing table**

   A → B , B → C , C → D ⋯ Y → Z

Private IP vs Public IP And NAT (Network Address Translation)



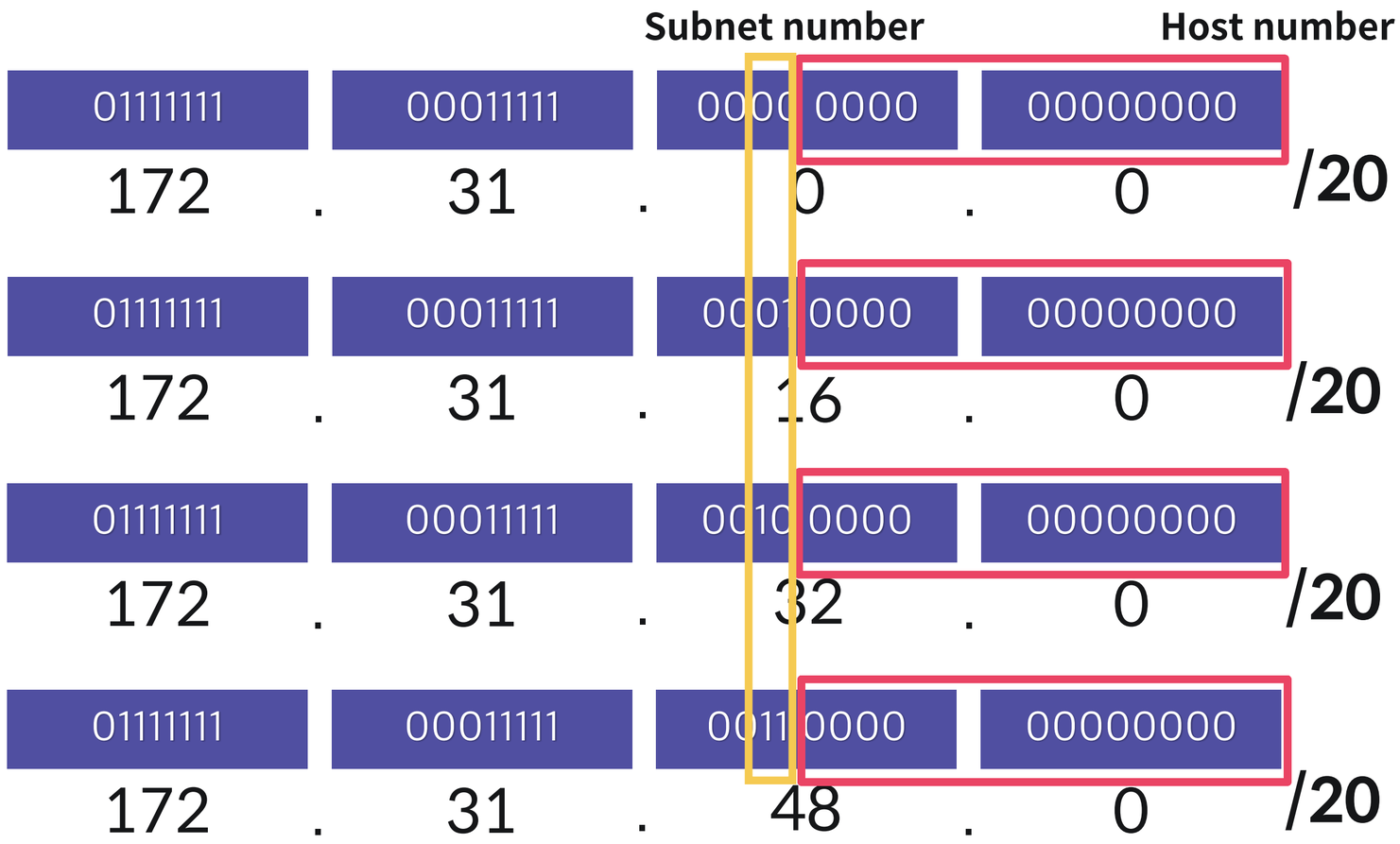
Private IP vs Public IP

- **Private IP** : Public IP의 절대적인 개수의 제한을 보완하기 위해 내부에서만 사용되는 IP
- **Public IP** : 인터넷상에서 직접적으로 사용되는 고유의 IP

NAT

- **NAT :** 여러 개의 Private IP를 가진 서버가 인터넷과 통신하기 위해 Public IP로 변환하는 작업

IP Subnet 동작 원리



Subnet 별 IP 대역 (CIDR 블럭)

172.31.0.0 ~ 172.31.15.255

172.31.16.1 ~ 172.31.31.255

172.31.32.1 ~ 172.31.47.255

172.31.48.1 ~ 172.31.63.255

Subnet : 네트워크 대역을 논리적으로 나누어
IP를 효율적으로 할당하고 네트워크 망 분리

요약

1. 우리가 익숙하게 사용하는 웹 브라우저 주소는 DNS 서버를 통해 IP 주소로 변환된다.

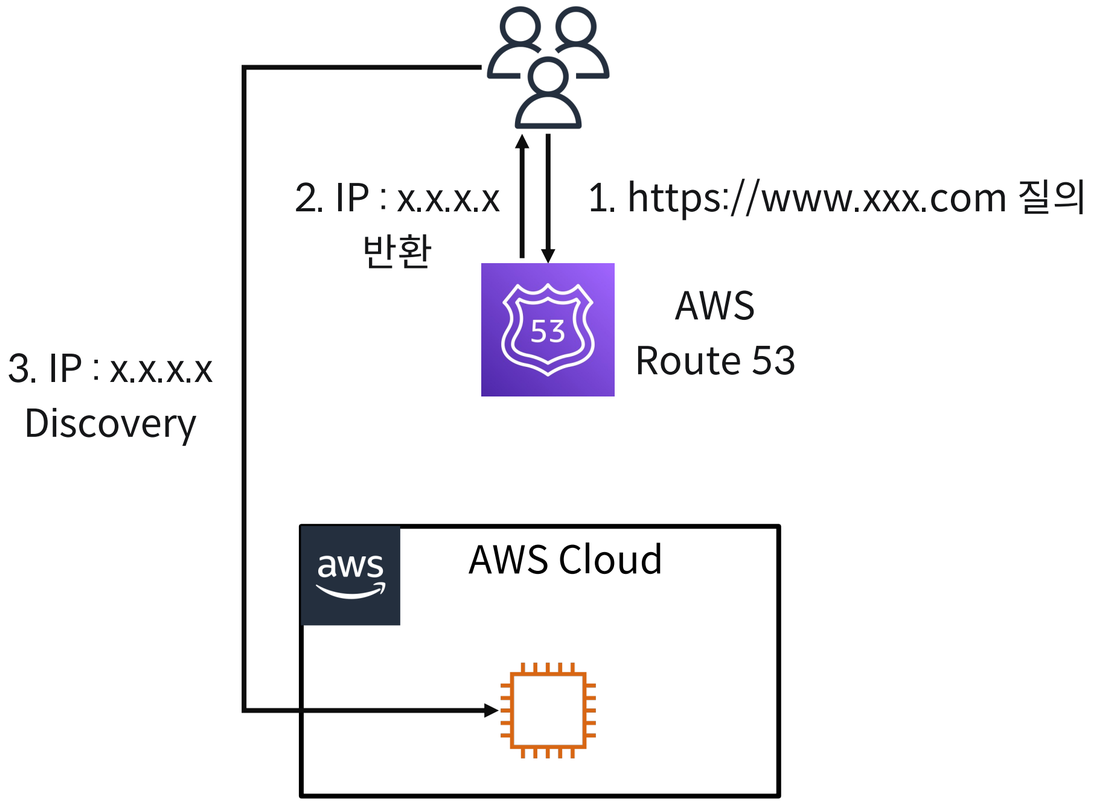
2. 패킷에는 전달하려는 데이터와 도착지 주소를 가지며, IP 라우팅을 통해 도착지로 전달된다.

3. 한정된 Public IP의 개수를 보완하기 위해 Private IP 와 NAT 개념이 사용된다.

02

AWS 네트워킹 2 : Route 53

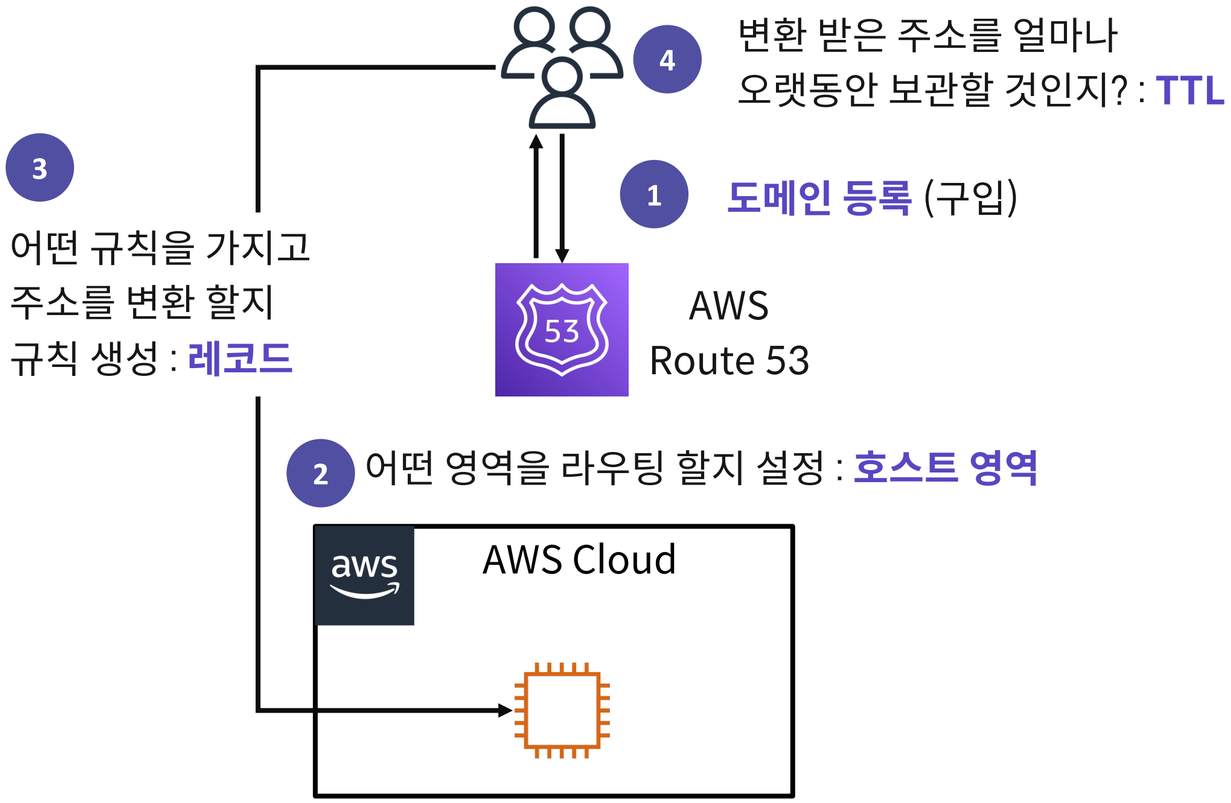
AWS Route 53 란 무엇인가요?



Route 53 : AWS 의 완전 관리형 글로벌 DNS 서비스

- 사용자에게 친숙한 Host name (ex. www.google.com) 을 AWS 서비스의 IP 주소로 변환 (라우팅) 하는 역할
- AWS 서비스가 정상적으로 동작하는 지 헬스 체크

AWS Route 53 구성 절차

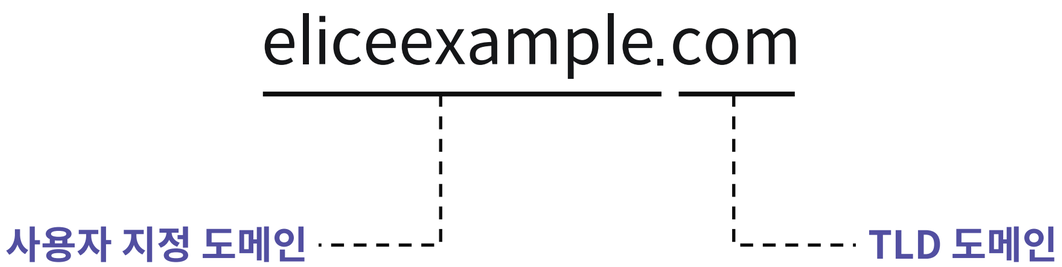


1. TTL 설정

Route 53 구성을 위한 기본 절차

1. 도메인 등록
2. 호스트 영역 설정
3. 레코드 설정

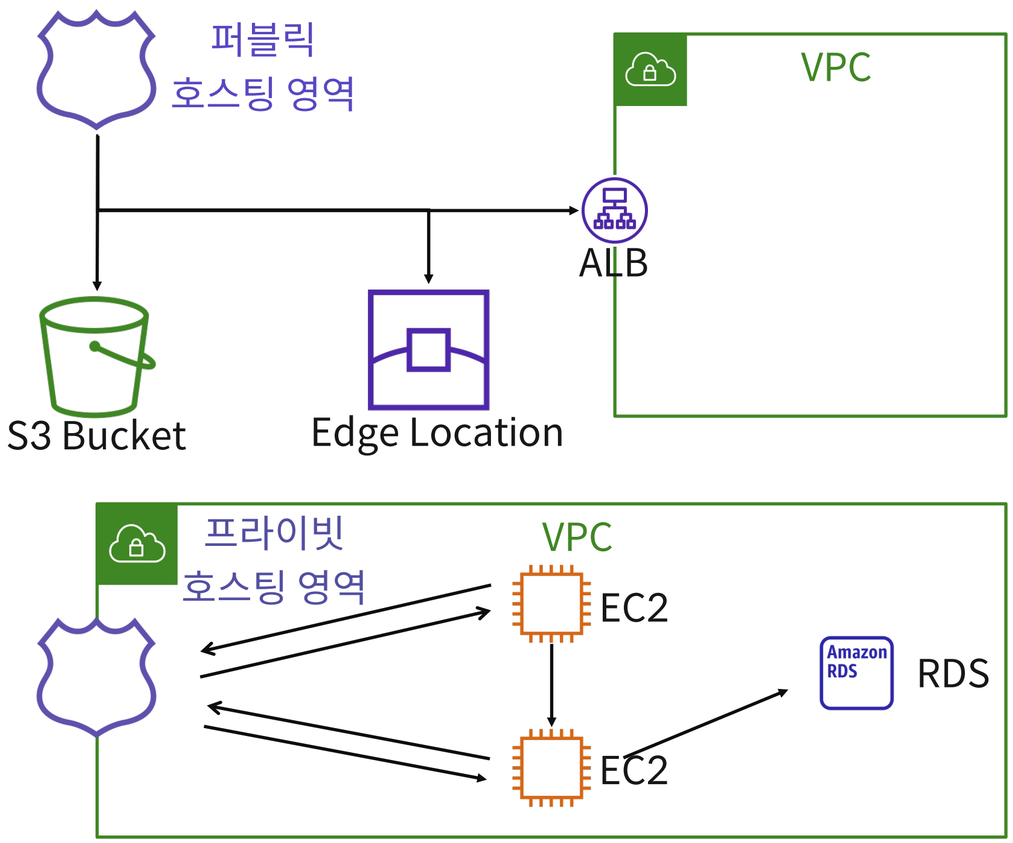
AWS Route 53 도메인 등록



1. 도메인 등록 규칙

- 등록되는 도메인은 전 세계적으로 유일해야 함
- 대표적인 TLD 도메인은 (com, org, co.kr 등) 이며, TLD 도메인별 비용 상이
- 사용자 지정 도메인은 문자, 숫자, 하이픈 (-) 을 포함할 수 있으며 64자 미만

AWS Route 53 호스팅 영역



2. 호스팅 영역 지정

- 하나의 독립적인 도메인 이름 (elice.io)을 가지며, 그 안의 여러 개의 서브 도메인 (academy.elice.io) 을 가지는 단위

- **퍼블릭 호스팅 영역** : 인터넷상에서 AWS 서비스에 대한 라우팅을 위한 영역
- **프라이빗 호스팅 영역** : VPC 내부의 AWS 서비스에 대한 라우팅을 위한 영역

AWS Route Record

레코드 이름	레코드 유형	값	TTL	라우팅 정책
www.elice.io	A	IP : x.x.x.x	300	단순 라우팅
academy.elice.io	A	IP : y.y.y.y	300	단순 라우팅
accounts.elice.io	A	IP : z.z.z.z	60	단순 라우팅

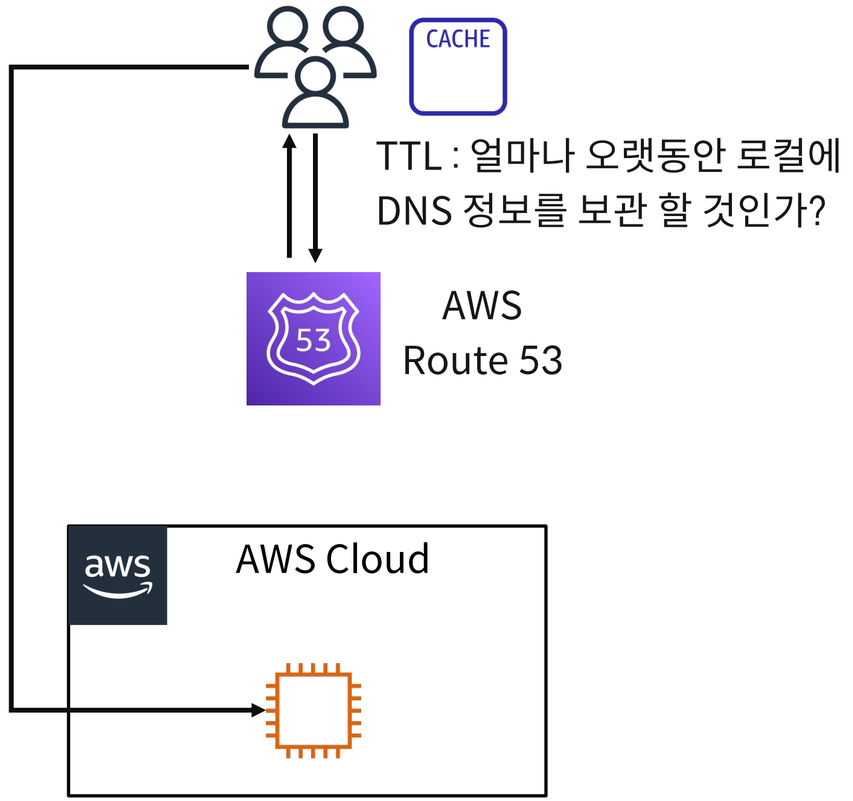
3. 레코드 구성하기

레코드는 하나의 도메인 영역에서 어떤 규칙으로 라우팅을 할 것 인가에 대한 정보를 가짐

- 레코드 유형

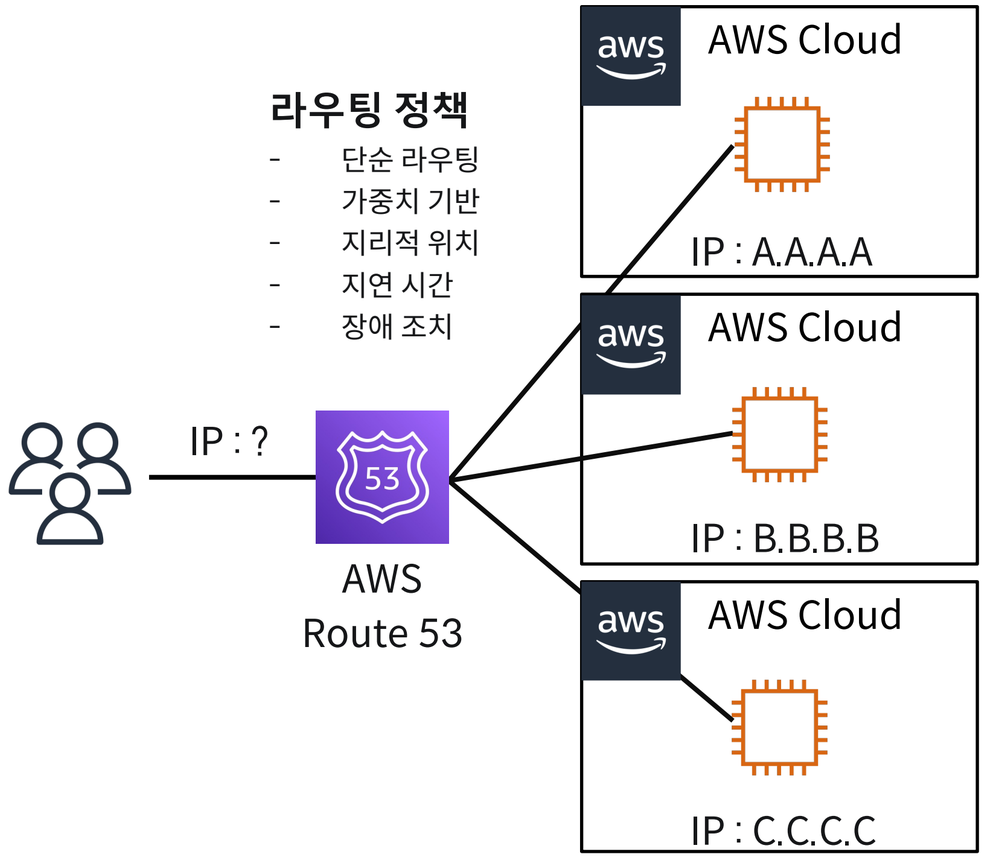
- A : 도메인 이름 ↔ IPv4 주소
- AAAA : 도메인 이름 ↔ IPv6 주소
- CNAME : 도메인 이름 ↔ 도메인 이름

AWS Route 53 TTL (Time to Live) & Routing 정책



Route 53 TTL

Route 53 에 접속하는 빈도를 줄여
효율적으로 라우팅 하기 위한 목적



Route 53 Routing 정책

DNS 쿼리를 어떤 방식으로
할지 에 대한 정책 정의

요약

1. AWS Route 53 는 AWS 의 관리형 DNS 서비스 이다.

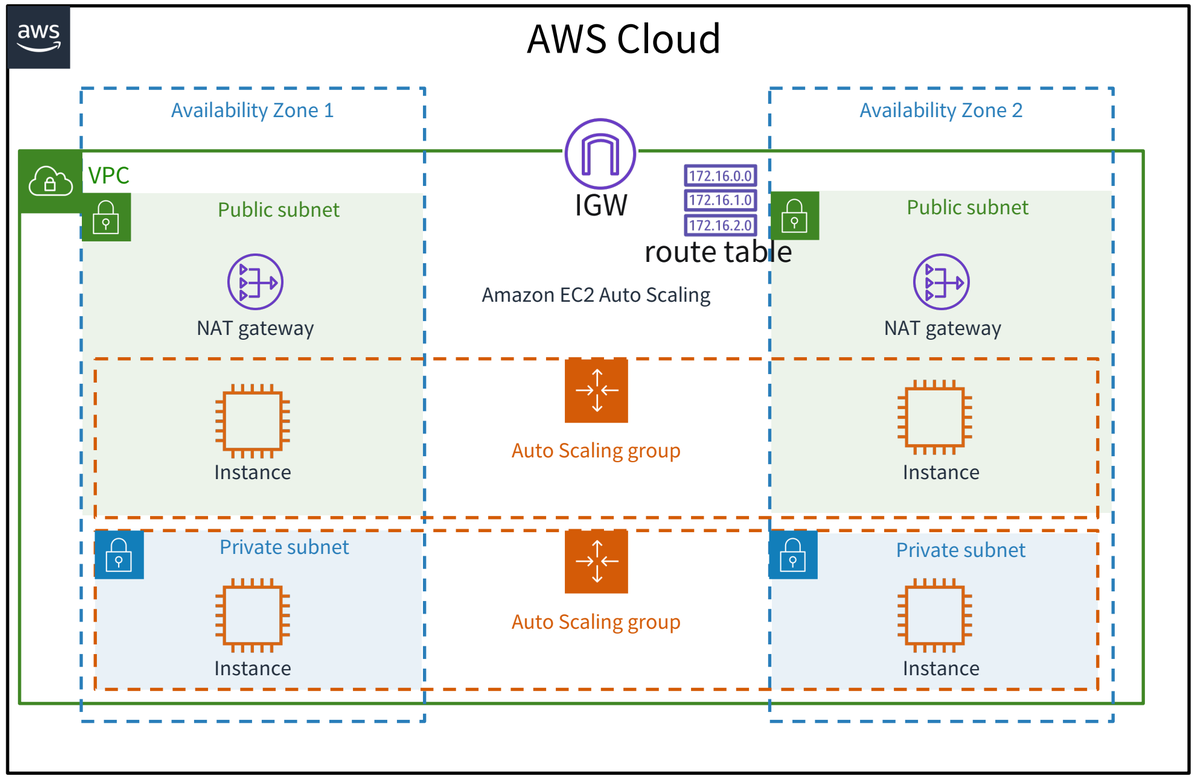
2. Route 53 의 동작을 위해 호스팅 영역 (어디를 대상으로)과 레코드 (어떤 규칙으로)의 지정이 필요하다.

3. 효율적인 DNS 쿼리를 위해 TTL과 라우팅 정책을 고려해야 한다.

03

AWS 네트워킹 3 : VPC (Virtual Private Cloud)

VPC 란 무엇인가요?



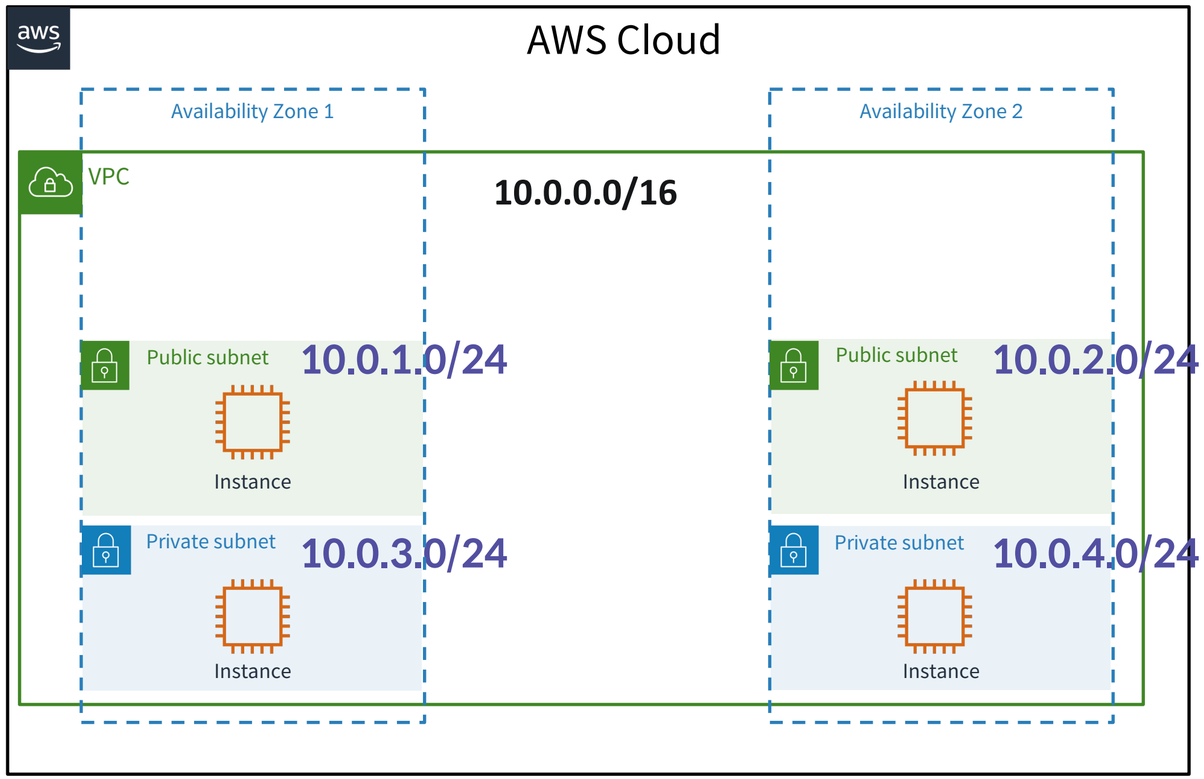
VPC (Virtual Private Cloud)

- AWS 환경에서 존재하는 가상의 네트워크
- Region 단위로 존재하여, 하나의 큰 IP CIDR Block 을 가짐.

주요 구성요소

- Subnet (Public / Private)
- Routing Table
- Internet Gateway
- NAT Gateway

Private Subnet vs Public Subnet

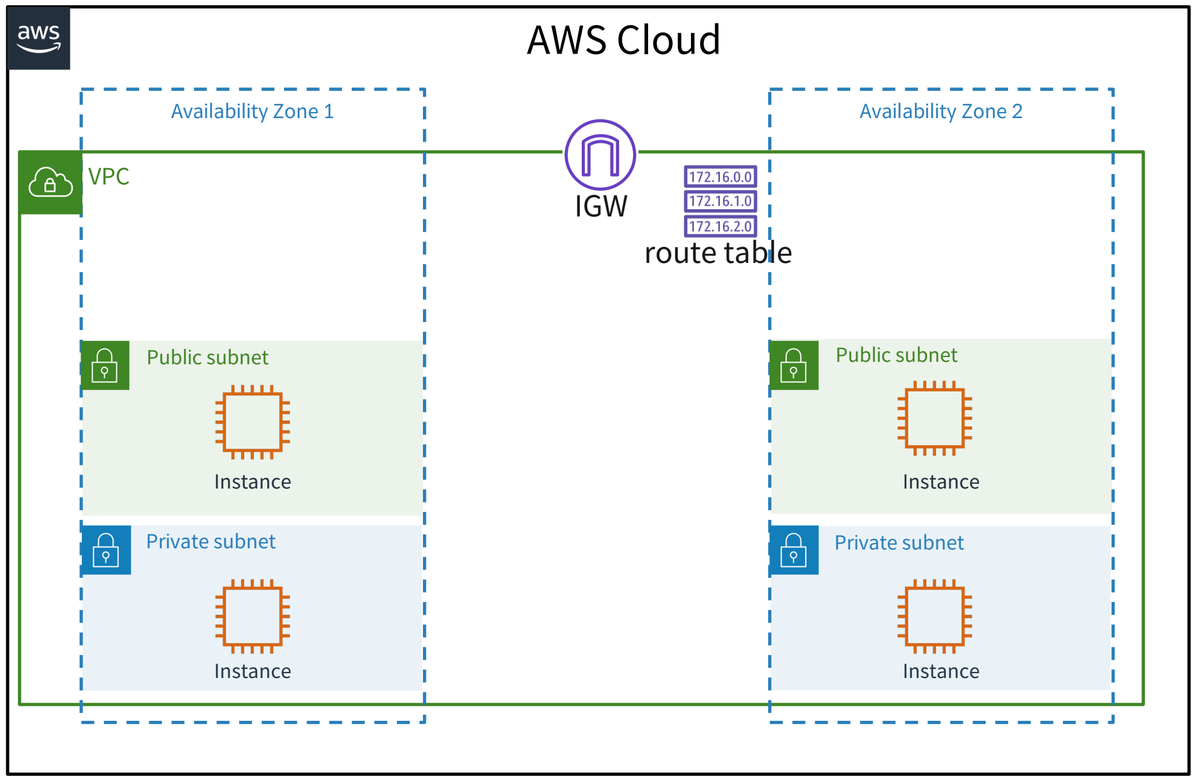


Subnet

- VPC 가 가지는 큰 IP CIDR 블럭을 Subnetting (서브네팅) 하여 용도, 목적별로 분리하여 실제 AWS 서비스가 배치되는 곳
- 가용 영역 (AZ) 단위로 서브넷 배포

Public Subnet : 인터넷과 직접적으로 연결되어 통신 가능한 서브넷 (인터넷 접속 가능한 EC2 혹은 ALB 배치)
Private Subnet : 인터넷과 완전히 격리된 네트워크 서브넷 (WAS 서버나 DB 서비스 배치)

IGW (Internet Gateway)과 Route table 은 무엇인가요?



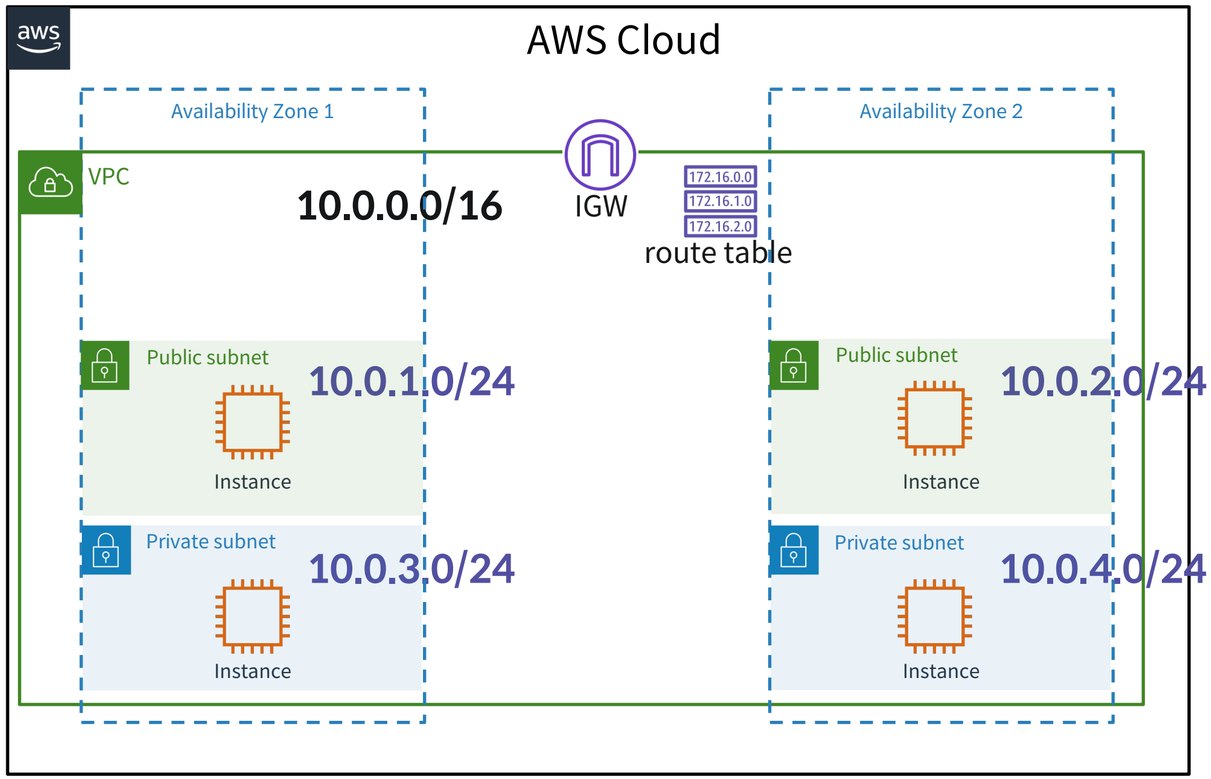
IGW (Internet Gateway)

- VPC 라는 격리된 AWS 네트워크 환경에서 인터넷 환경과 마주 보고 있는 Gateway

Route Table

- VPC 내의 서브넷들이 보내려는 주소로 가기 위해 어디로 향해 보내야 하는지 정보를 가지는 테이블
- 도착지 (Destination) 와 대상 (Target) 으로 구성

VPC 구성해보기 : VPC 및 Subnet 분리



1. VPC 생성
- name : elice-vpc01
- CIDR Block : 10.10.10.0/16

2. Subnet 생성 (Public)
- name : elice-public-subnet01~02
- 가용 영역 : ap-northeast-2a / 2b
- CIDR Block : 10.10.1.0/24 ~ 10.10.2.0/24

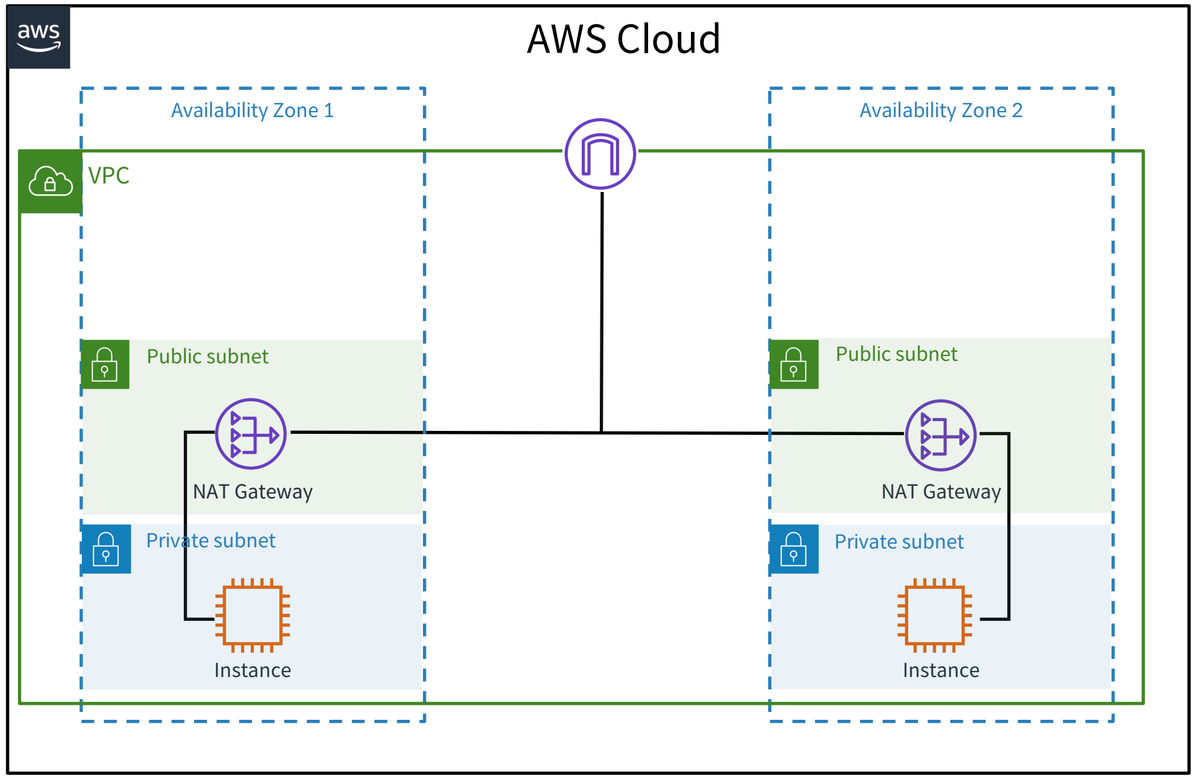
3. Subnet 생성 (Private)
- name : elice-private-subnet03~04
- 가용 영역 : ap-northeast-2a / 2b
- CIDR Block : 10.10.3.0/24 ~ 10.10.4.0/24

4. IGW 생성
- name : elice-igw
- VPC attach : elice-vpc01

5. route table 생성 (Public)
- name : elice-public-route01
- VPC : elice-vpc01
- 서브넷 연결 :elice-public-subnet01~02
- 라우팅 편집 : 대상 : 0.0.0.0/0 → elice-igw

6. route table 생성 (Private)
- name : elice-private-route01
- VPC : elice-vpc01
- 서브넷 연결 :elice-private-subnet03~04

NAT Gateway 란 무엇인가요?



NAT Gateway

- Private Subnet 에 위치하는 서비스들을 인터넷과 통신하기 위해 Public 주소로 변환해 주는 관리형 NAT 서비스
- AZ 내 서브넷 단위로 배포
- 트래픽 흐름 : Private Subnet Instance → NAT Gateway → IGW
- 고 가용성을 위해 AZ 별로 각각 배포
- NAT 당 고유한 Public IP 주소 할당

요약

1. VPC 는 AWS 내의 독립적인 가상의 네트워크이다.

2. VPC 내부는 서브넷 단위로 분리되며, AWS 서비스는 실질적으로 서브넷에 위치한다.

3. 서브넷에 위치한 EC2 인스턴스는 Private IP 혹은 Public IP 를 가지며, 라우팅 테이블에 의해 통신한다.

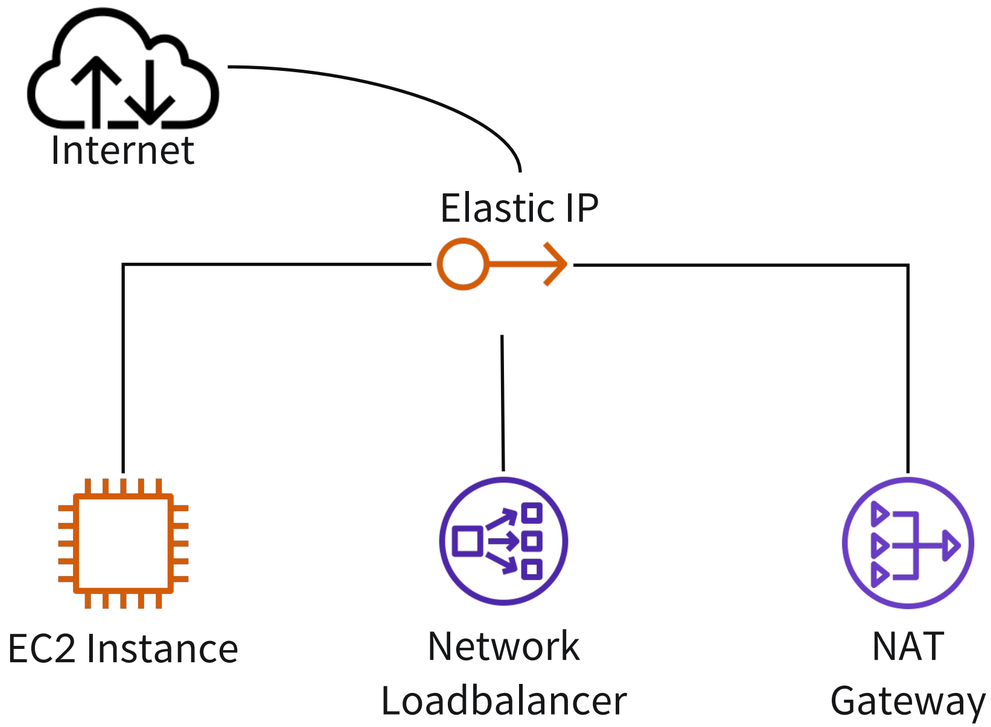
4. 인터넷망과 연동을 위해 IGW, NAT 등이 사용된다.

04

AWS 네트워킹 4 : VPC 중급

/* elice */

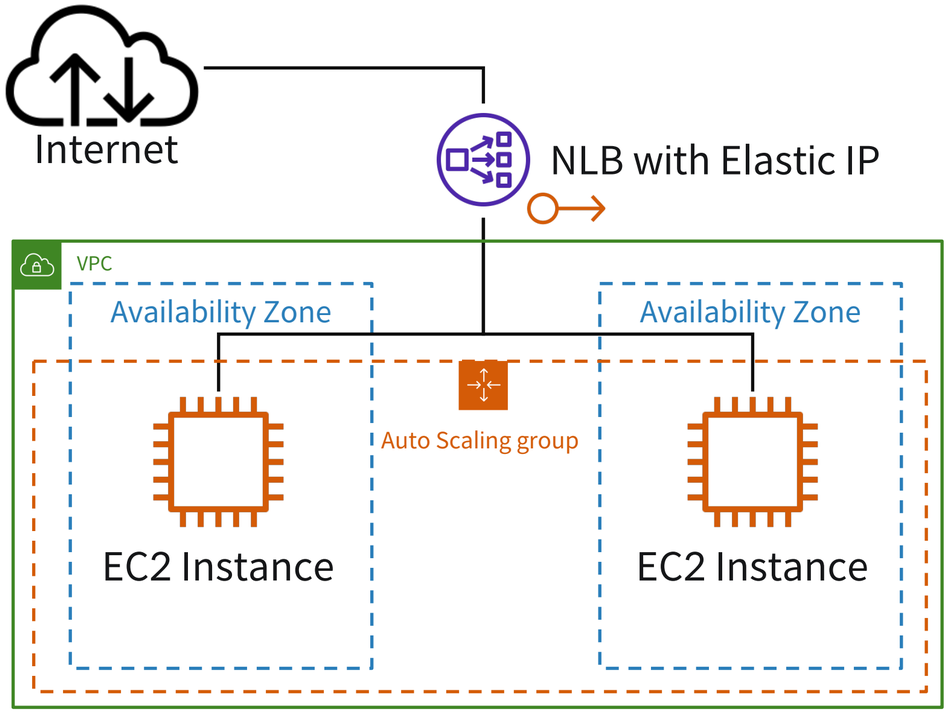
AWS Elastic IP : AWS Public IP 할당하는 방법



Elastic IP (EIP)

- 탄력적 IP 는 인터넷과 직접적으로 통신할 수 있게 해주는 AWS 의 Public IP = 전 세계적으로 유일한 IP
- 계정 당 최대 5개까지 생성 가능
- 보안을 위해 직접적으로 노출되지 않도록 DNS 나 Load Balancer 사용

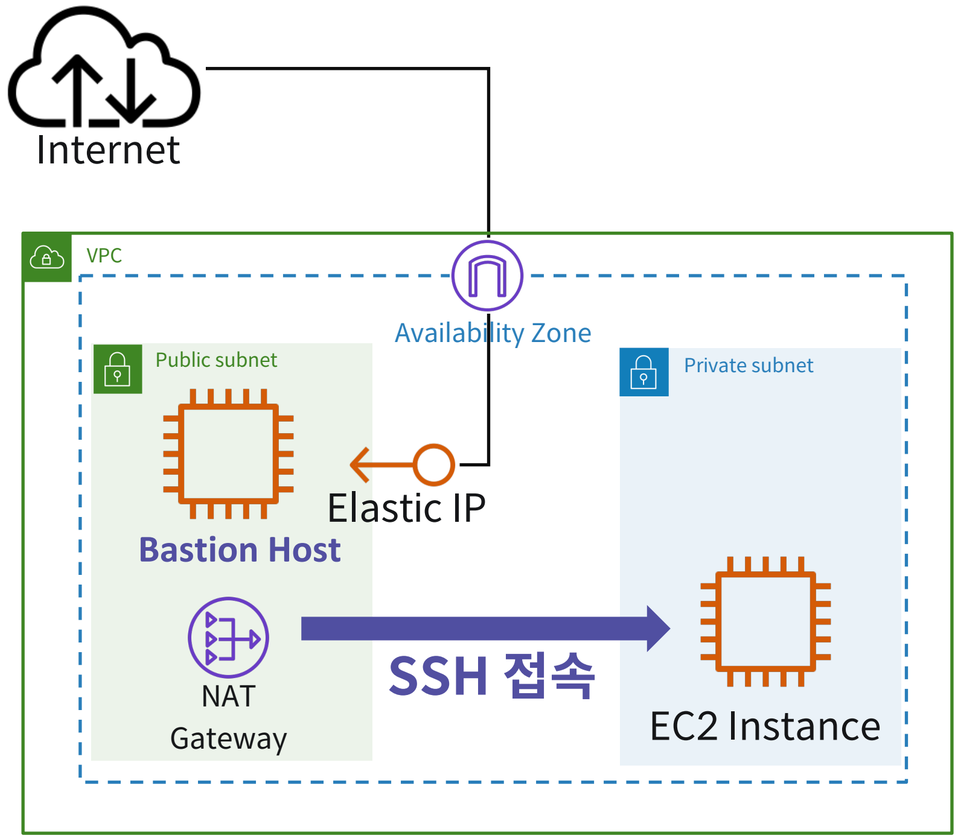
Public IP 활용 예시 1 : 외부 노출이 필요한 고 가용성 인스턴스



고 가용성 인스턴스

- IP 통신이 필요한 인스턴스의 대표적인 고 가용성 구성
- Network Load Balancer 는 고유의 Public IP 를 가지며, EC2 인스턴스는 Private IP 를 가지고도 외부로 통신 가능

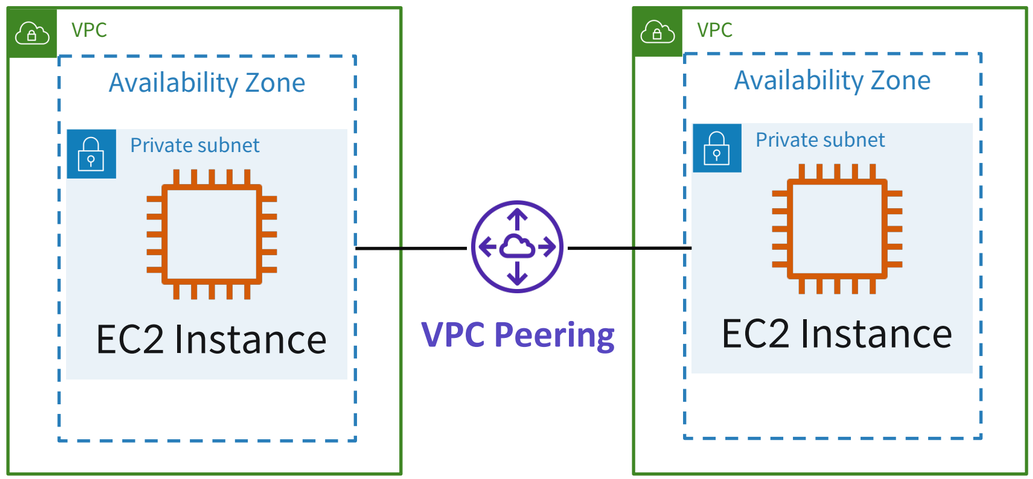
Public IP 활용 예시 2 : Bastion Host



Bastion Host

- Bastion Host 는 퍼블릭 혹은 프라이빗 서브넷에 위치한 EC2 인스턴스를 안전하게 접속하도록 돕는 **EC2 Instance**
- Private Subnet 에 위치한 EC2 인스턴스는 NAT 를 통해 외부 통신
- Bastion Host 는 NAT 를 통해 해당 인스턴스를 SSH 접속

VPC Peering : VPC 간의 연동 하는 방법



VPC Peering

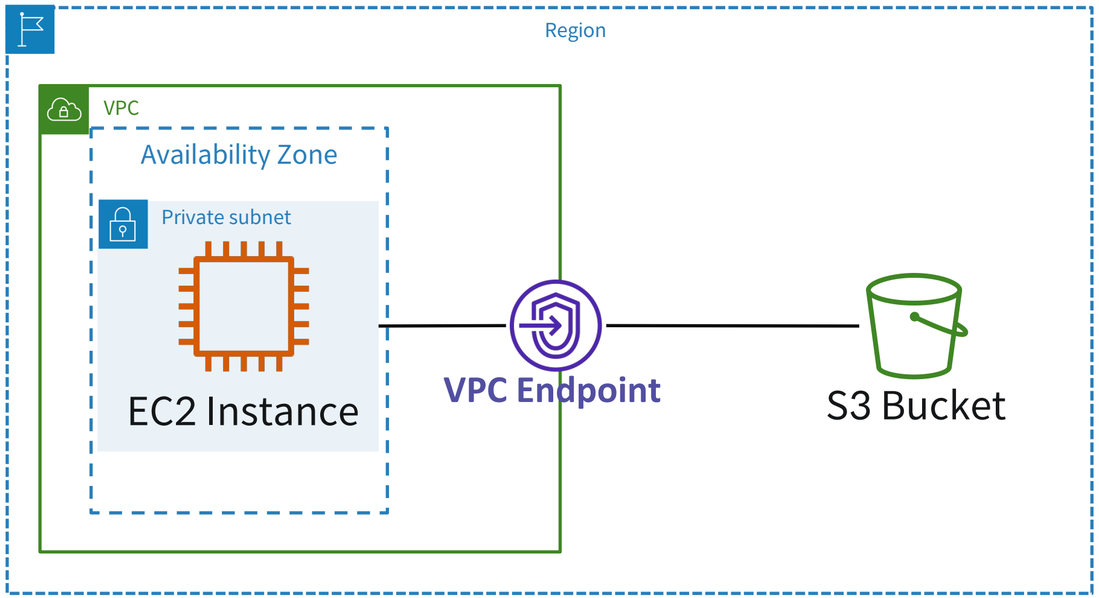
두 개의 VPC 의 트래픽을 라우팅 할 수 있도록 하는 VPC 간의 네트워킹 연결

VPC Peering 사용 예시

- 보안 목적 : Peering 을 통해 연결된 VPC 는 IGW 구성을 하지 않고 완전히 인터넷 영역과 차단된 환경 구성 가능

- 유연한 서비스 분리 : 백엔드 서비스와 같이 인터넷 서비스와 직접적인 연결이 필요 없는 서비스의 경우, NAT 나 Public IP 구성없는 별도 환경 구성 가능

VPC Endpoint : 프라이빗 네트워크로 AWS 서비스 접속하는 방법

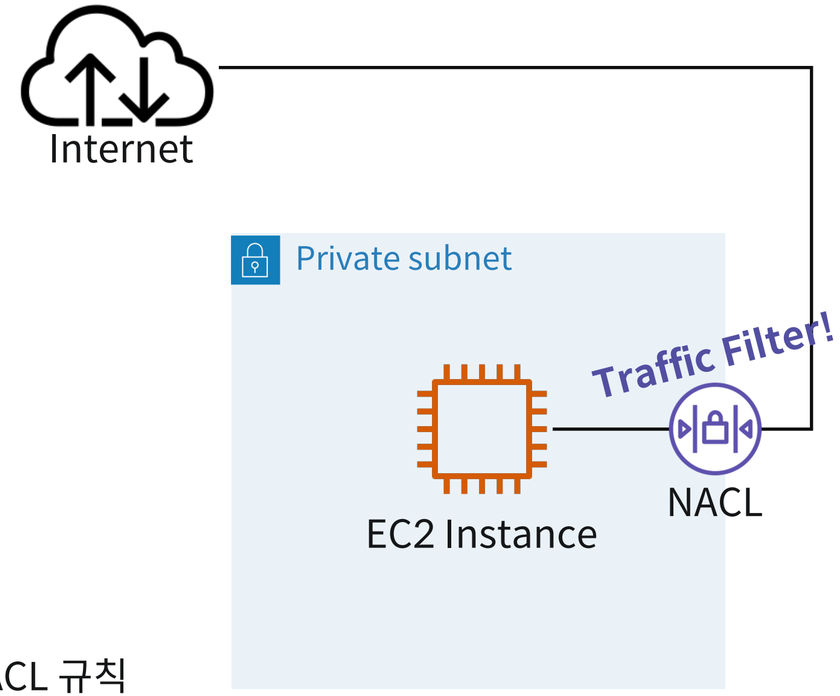


VPC Endpoint

- VPC 내 존재하는 서비스와 Region 에 존재하는 서비스와 통신하기 위해서는 인터넷을 거쳐야 함

- VPC Endpoint 는 인터넷을 거치지 않고, AWS 프라이빗 네트워크를 사용하여 VPC 내 서비스와 Region 내 서비스와 연결

NACL (Network Access Control List) 이란 무엇인가요?



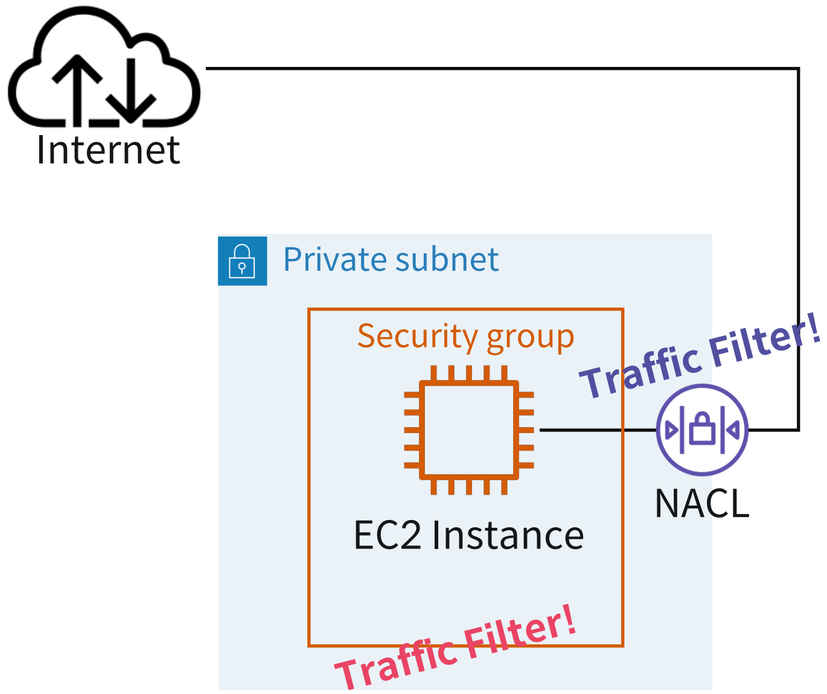
NACL 규칙

규칙 번호	유형	프로토콜	포트	소스	허용 / 거부
100	모든 트래픽	TCP	모두	0.0.0.0/0	허용

NACL (Network Access Control List)

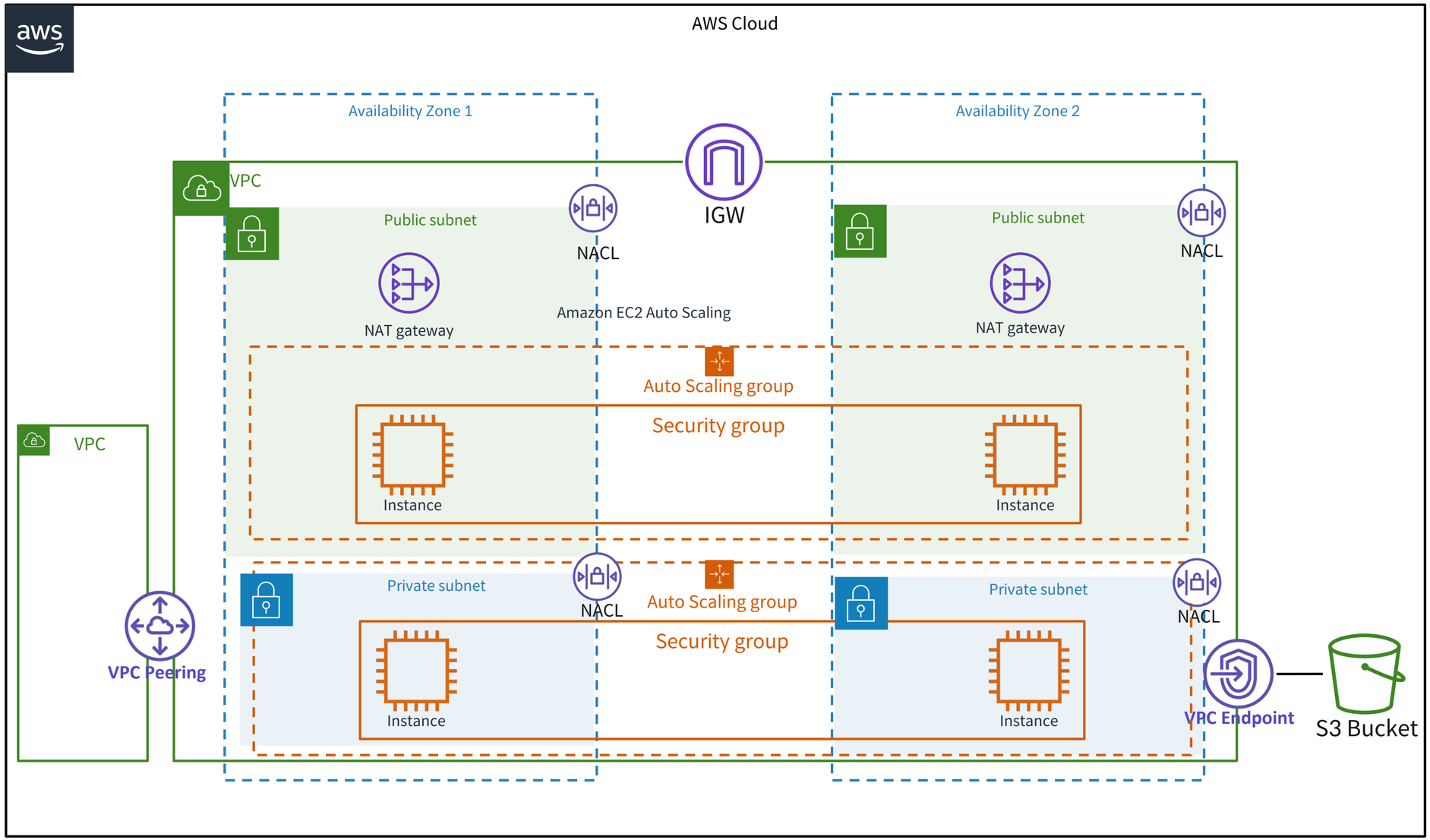
- NACL 은 서브넷 레벨에서 트래픽을 관리하는 방화벽
- 서브넷 생성 시 Default NACL 이 자동으로 생성되며, 모든 트래픽은 Allow
- 방화벽 규칙은 IP, Port, Protocol 로 제어
- 규칙 번호를 가지며, 작은 숫자대로 순차적으로 검사

보안 그룹 (Security Group) vs NACL



보안 그룹	NACL
인스턴스 레벨에서 적용	서브넷 레벨에서 적용
허용 규칙만 허용	허용 및 거부 규칙 지원
상태 저장 : 규칙에 관계없이 반환 트래픽이 자동으로 허용	상태 비 저장 : 반환 트래픽이 규칙에 의해 명시적으로 허용
트래픽 허용 여부를 결정하기 전에 모든 규칙을 평가	트래픽 허용 여부를 결정할 때 번호가 낮은 규칙부터 순서대로 규칙 처리
인스턴스 시작 시, 보안 그룹을 인스턴스에 지정해야만 보안 그룹 규칙이 적용됨	연결된 서브넷의 모든 인스턴스에 자동 적용

VPC 기능을 활용한 아키텍처



요약

1. VPC 환경에서 Public IP 혹은 NAT + Bastion Host 를 통해 안전하게 AWS 서비스를 접근할 수 있다.

2. VPC 간 네트워킹으로 VPC Peering, VPC 와 외부서비스 간 네트워킹으로 VPC Endpoint 가 있다.

3. VPC 보안을 위해 Security Group 과 NACL 이 활용한다.

05

AWS 환경에서 탄탄한 웹 서버 구축하기

웹 서버를 구축할 시 고려해야 할 점

성능 : 사용자가 만족할 수준의 충분한 성능을 제공하고 있는가?

Computing, Database, Storage, CloudFront

안정성 : 내 서비스는 여러 명이 들어와도 충분히 수용 가능한가?

Instance Type, Loadbalancer, Auto scaling, Read replica

고 가용성 : 하나의 서비스가 문제가 생겨도 지속적인 서비스가 가능할까?

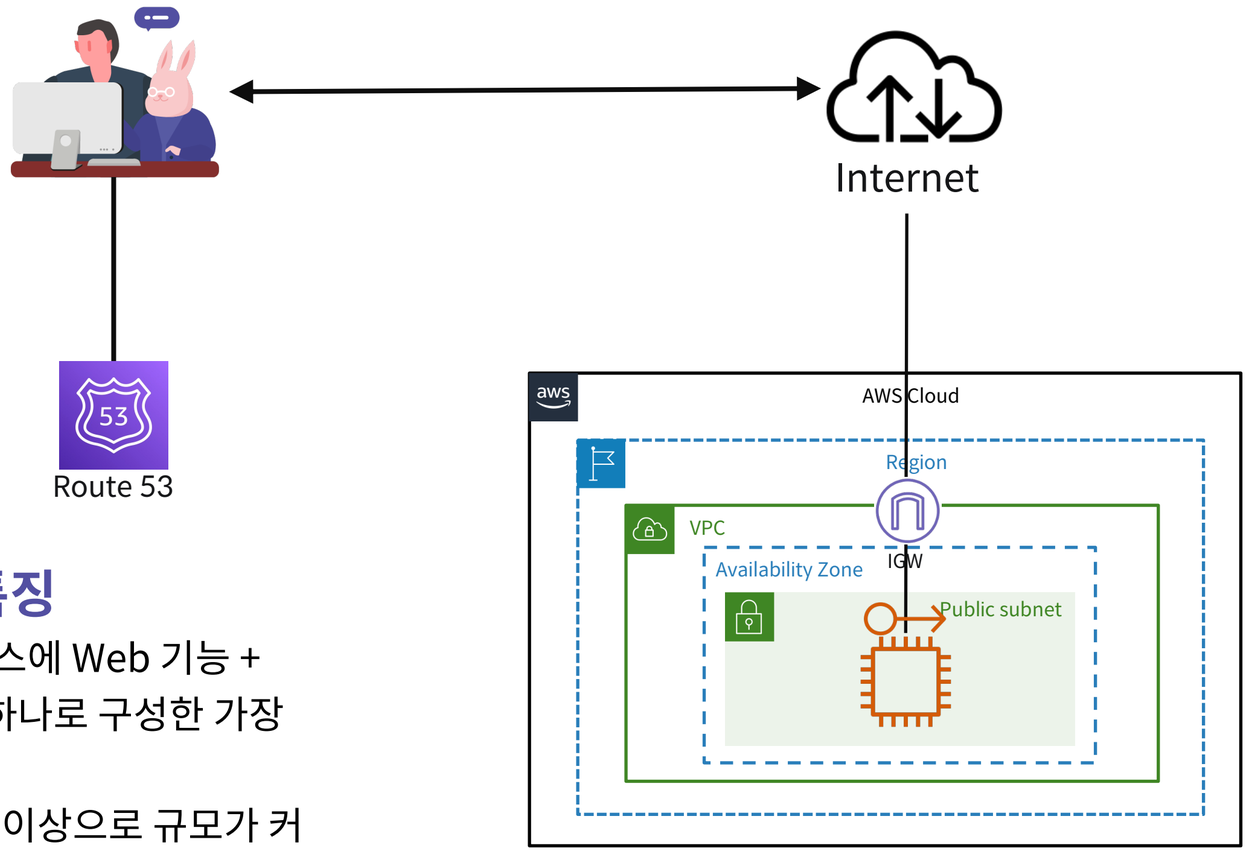
Region, AZ

보안 : 구성한 서비스들이 악의적인 접근으로부터 충분히 보호 받고 있는가?

IAM, Security group, Subnet, NACL, Encryption

AWS 서비스 활용은 블록 조립

1. 첫 서버 구축하기 : 단일 EC2 인스턴스



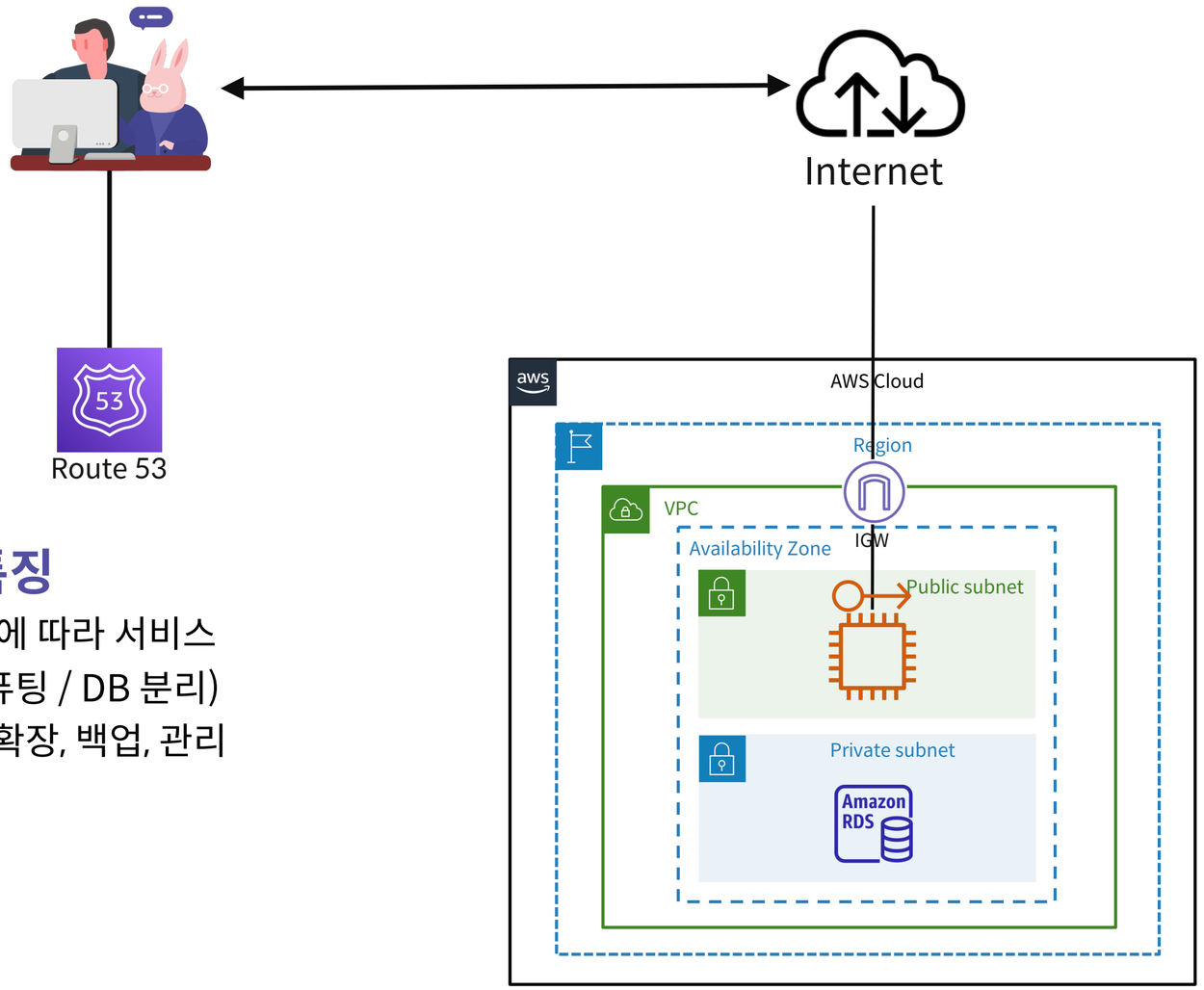
아키텍처 특징

- EC2 인스턴스에 Web 기능 + DB 기능을 하나로 구성한 가장 단순한 구조
- 일정 트래픽 이상으로 규모가 커졌을 때, 수용하기 어려움

서비스

- EC2 인스턴스
- Elastic IP
- VPC (Subnet)
- Route 53

2. 서비스 세분화 하기 : RDS 구성



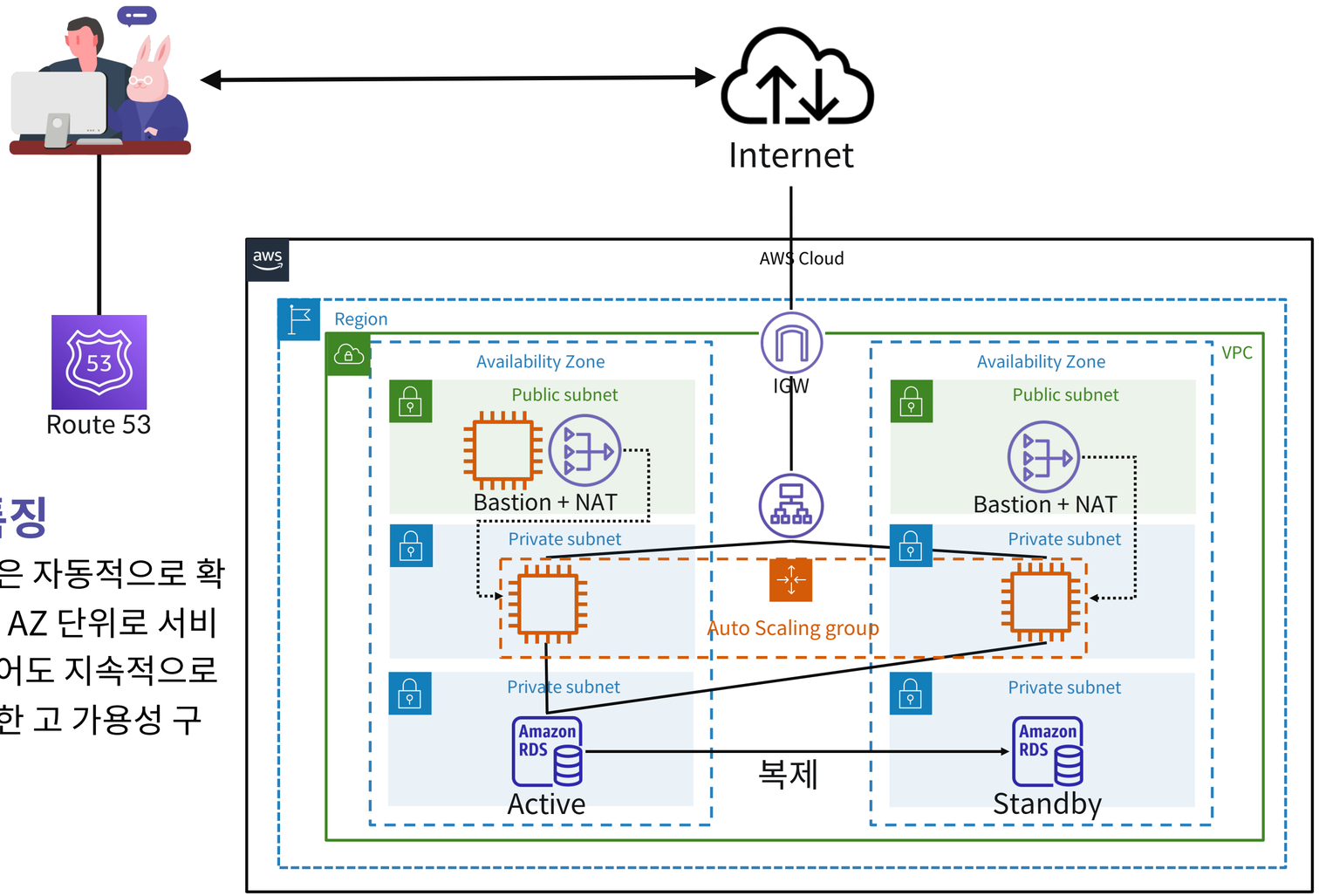
아키텍처 특징

- 기능과 역할에 따라 서비스를 분리 (컴퓨팅 / DB 분리)
- DB의 성능, 확장, 백업, 관리 측면 개선

서비스

- EC2 인스턴스
- Elastic IP
- VPC (Subnet)
- Route 53
- RDS

3. 가용성 고려한 구성 : 다중 인스턴스 + Multi AZ



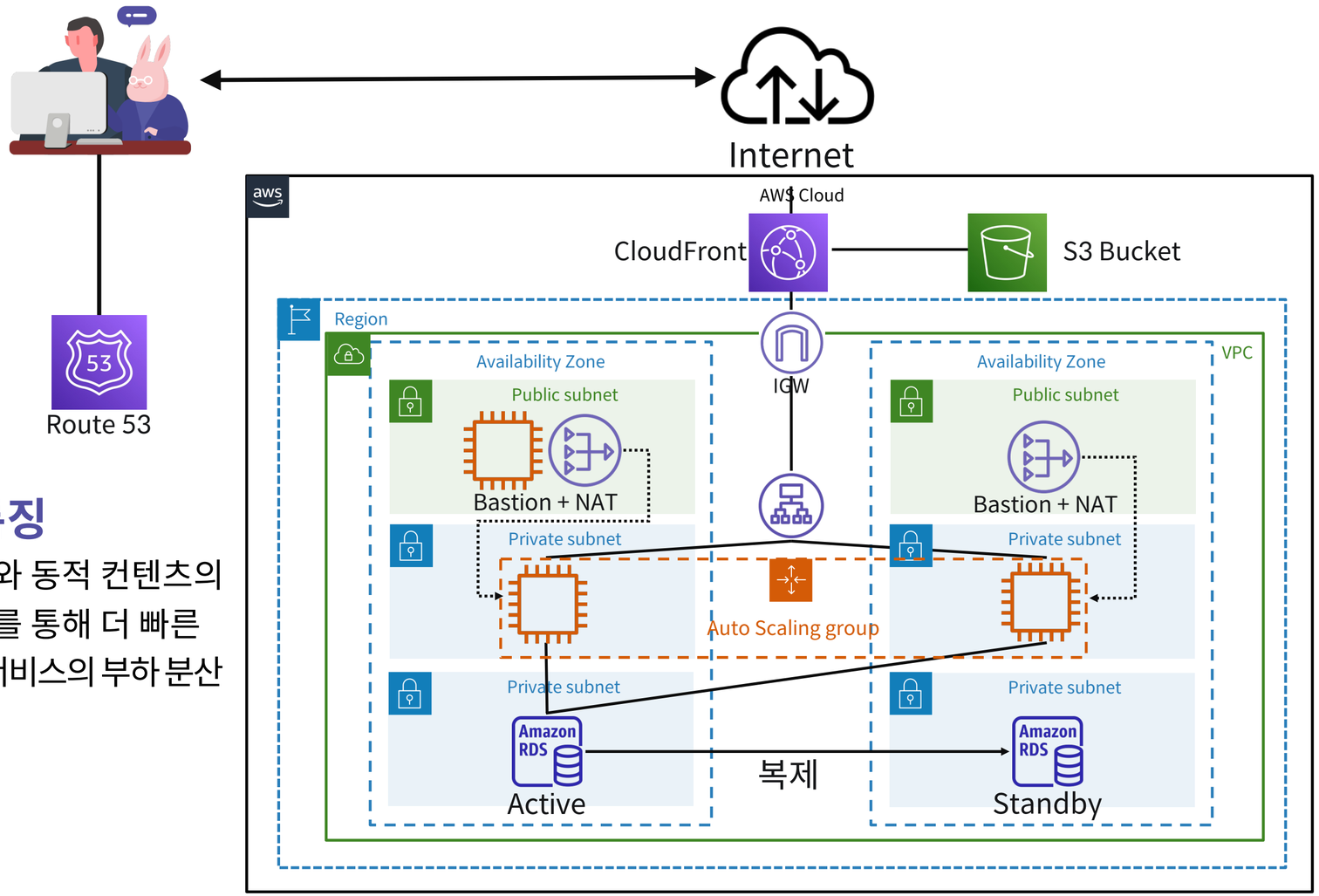
아키텍처 특징

- 각 서비스들은 자동적으로 확장 가능하며, AZ 단위로 서비스가 중단되어도 지속적으로 서비스 가능한 고 가용성 구성

서비스

- EC2 인스턴스
- Elastic IP
- VPC (Subnet)
- Route 53
- RDS
- Bastion + NAT
- Auto Scaling
- ELB
- Multi AZ

4. 트래픽 분산 구성 : CloudFront + S3



아키텍처 특징

- 정적 컨텐츠와 동적 컨텐츠의 트래픽 분리를 통해 더 빠른 응답시간과 서비스의 부하 분산

서비스

- EC2 인스턴스
- Elastic IP
- VPC (Subnet)
- Route 53
- RDS
- Bastion + NAT
- Auto Scaling
- ELB
- Multi AZ
- CloudFront
- S3

출처 : https://docs.aws.amazon.com/ko_kr/whitepapers/latest/web-application-hosting-best-practices/an-aws-cloud-architecture-for-web-hosting.html

요약

1. 탄탄한 웹 서버 아키텍처를 만들기 위해서는 성능, 안정성, 가용성, 보안, 비용 등이 고려되어야 한다.

2. AWS 서비스 아키텍처링은 블록 조립과 같다. 각 서비스의 특성을 충분히 이해하고 구현할 서비스에 적합한 AWS 서비스를 활용해야 한다.

크레딧

/* elice */

코스 매니저
임승연

콘텐츠 제작자
Jason

강사
Jason

감수자
장석준

디자이너
강혜정

연락처

TEL

070-4633-2015

WEB

https://elice.io

E-MAIL

contact@elice.io